Pattern Recognition, Vol. 24, No. 5, pp. 453–464, 1991
Printed in Great Britain

0031-3203/91 $3.00 + .00
Pergamon Press plc
© 1991 Pattern Recognition Society

SKELETONIZATION OF ARABIC CHARACTERS USING CLUSTERING BASED SKELETONIZATION ALGORITHM (CBSA)

SABRI A. MAHMOUD,* IBRAHIM ABUHAIBA* and ROGER J. GREEN†
* Computer Engineering Department, CCIS, King Saud University, Riyadh, Saudi Arabia; † Electrical Engineering Department, University of Bradford, Bradford, U.K.

(*Received* 29 *May* 1990; *in revised form* 11 *October* 1990; *received for publication* 19 *October* 1990)

Abstract—Character skeletonization is an essential step in many character recognition techniques. In this paper, skeletonization of Arabic characters is addressed. While other techniques employ thinning algorithms, in this paper clustering of Arabic characters is used. The use of clustering technique (an expensive step) is justified by the properties of the generated skeleton which has the advantages of other thinning techniques and is robust. The presented technique may be used in the modeling and training stages to reduce the processing time of the recognition system.

Arabic characters Skeletons Thinning Fuzzy clustering ISODATA
Character recognition

1. INTRODUCTION

Character recognition is an area of pattern recognition that has been subjected to considerable research work during the past three decades.[1–5] The recognition and processing of type- (or hand-) written text is necessary to improve man–machine communication and is useful in office automation.

Skeletonization of characters is a primary stage in many character recognition techniques.[6–8] Pavlidis[9] defines the skeleton as the set of points P which has semi-equal distance from two or more points of the pattern contour. Ideally, the original character should be thinned to its medial axis. The skeleton representation is often an efficient method for expressing the structural relationship of the major components of an object. The advantages of skeletons are the reduction in the required memory space for storing the essential structural information of the patterns, the simplification of the data structures required in processing the pattern, and the reduction in the required processing time. In general, in character recognition, skeletons are obtained by thinning operations or distance transforms.

Many algorithms have been proposed in the past to extract skeletons.[9–14] Dinneen[10] proposed a thinning algorithm which produces skeletons with thin vertical lines and thick horizontal lines. So, the final skeleton is not consistently of unity width. Rosenfeld and Pfaltz[11] presented an algorithm which produces skeletons that may not be connected. However, this algorithm is one of the fast algorithms. Hilditch[12] described a thinning algorithm in which the pixels of the outside layer are removed iteratively until no more pixels can be removed. Remaining pixels constitute the object skeleton. This algorithm yields connected skeletons which are not sensitive to contour noise. Although all the tests necessary before a dark point could be deleted from a pattern were discussed, the approach was not presented formally. The algorithm of Udupa and Murthy[13] is not very robust as it requires each line to maintain a constant width over at least 6 pixels. Pavlidis[9] presented a thinning algorithm that gives spurious tails, and the strokes of the skeleton may have a thickness of more than 1 pixel.[14] The safe point thinning algorithm (SPTA) of Naccache and Shinghal[14] prevents excessive erosion and gives skeletons that retain the shape information of the original pattern, and do not have spurious tails assuming a smoothed input pattern. However, the skeleton suffers from bending of straight lines at "T" junctions, forming more of a "Y" shape. SPTA was compared with 14 other thinning algorithms and was shown to be the fastest. However, Pavlidis[9] has the same processing time. The SPTA algorithm calculations can be pipelined, such that minimal amount of memory is required.

Current skeleton algorithms generate skeletons that may not be unique and are sensitive to noise such that the final skeletons may be inaccurate. In this paper, a clustering based skeletonization algorithm (CBSA) is presented. This algorithm is based on clustering the character image. The adjacency matrix of the clusters is found and the skeleton is generated. An algorithm to refine the skeleton which removes insignificant vertices is applied. The resulting skeleton represents the character with fewer vertices and edges.

The organization of the paper is as follows. Section 2 describes the characteristics of the Arabic language script. Section 3 presents a skeletonization technique

(CBSA) that is based on the fuzzy ISODATA clustering technique. The initial assignment strategies and a skeleton reduction algorithm are also discussed. Section 4 details the experimental results addressing the presented algorithms. A comparison between CBSA, SPTA and Pavlidis[9] algorithms is included. Finally, the concluding remarks are given in Section 5.

2. CHARACTERISTICS OF ARABIC TEXT

The following are the general characteristics of Arabic text.

(1) The Arabic language is cursive (similar to cursive Latin script) and is written from right to left. Generally, an Arabic word consists of one or more connected segments, each consisting of one or more characters. The discontinuities between word segments is a result of the segments ending in characters that are not connectable from the left side with the succeeding character. The cursive nature of the Arabic language makes it difficult to separate a cursive word into characters.

(2) An Arabic character can have different shapes depending on its position in the word (beginning, middle, end or alone). This increases the complexity of the recognition of Arabic text.

(3) Arabic characters vary in size, even with using the same font of typewritten text.

(4) Many of the Arabic characters have dots which are positioned at a suitable distance above or below the character. Dots can be single, pairs or triples. Different Arabic characters can have the same body and differ in the number of dots identifying them.

(5) Arabic language uses another type of special character as short vowels which are referred to as diacritic. Although a different diacritic on the same character could lead to different words, an Arabic reader is trained to deduce the meaning of undiacriticized text from the context. When diacritics are used they appear above or below the characters and they are viewed as isolated entities. Some of the Arabic characters use special marks to modify the letter accent, such as Hamza(ء) and Madda(~), which are positioned at a certain distance above the character.

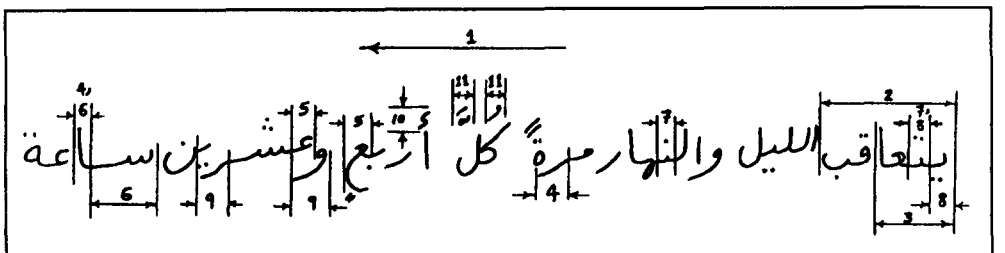
(6) Some of the Arabic characters may overlap with the neighboring characters. The degree of overlap can vary according to the typeface, the typewriter design, or the writer habits.

Figure 1 shows examples of the above properties.

In handwritten text large variations in the characters shape are expected, making the recognition stage more difficult. This is a result of the writing habit, style, education, mood, health and other conditions of the writer; in addition to other factors such as the writing instrument, writing surface and scanning methods. In order to obtain a reliable Arabic recognition system, the large variations in character writing type- (or hand-) written should be considered. So, a robust general model which can tolerate these variations and the special characteristics of the Arabic language is required.

3. SKELETONIZATION OF ARABIC CHARACTERS

Few researchers addressed the skeletonization of Arabic characters.[15-17] Due to the special characteristics of Arabic text, an Arabic recognition system is not a direct implementation of the recognition techniques used for other languages such as Latin and Chinese. For example, applying the thinning algorithms for Latin to an Arabic character with a single dot will produce a skeleton with a straight line for the dot. A dot is better modeled as 1 pixel



Notes:
(1) Direction of flow of Arabic writing (right to left). (2) One Arabic word consisting of two connected (cursive) segments. (3) A cursive segment consisting of four characters. (4) Characters which are not connectable from the left side with the succeeding character. (5) The same character with two different shapes dependent on its position in the connected portion. (6) Two different characters with two different sizes. (7) Two different characters with the same body, but with different number of dots. (8) Two different characters with the same character body and the same number of dots, but the position of dots differs for each. (9) Characters overlap. (10) A special mark "Hamza" is used to modify the character accent. (11) Example of diacritics: This word without diacritics can be read to mean one of the following: "every", "eat" and "to be tired". With the diacritics shown it means "every".

Fig. 1. An example of the characteristics of Arabic text.

and not as a short line. This has significance in handwritten Arabic text where two dots are often written as a short line. The thinning of a short line and a single dot will produce nearly the same skeleton using the above thinning algorithms. Hence, the character identity is ambiguous, as different number of dots indicate different characters.

A character skeleton can be viewed as a set of adjacent segments S_i, $i = 0, 1, \ldots, c-1$, where c is the number of segments. Each S_i is a group of adjacent pixels of the character image. The set of segments S_i and their relative location and adjacency to each other constitutes the character skeleton, which carries the necessary information for recognition.

There are several methods by which given data can be segmented into natural groupings. In this paper, the fuzzy ISODATA clustering technique is applied. The data are clustered into a pre-determined number of clusters. A number of pixels of the original character image are assigned to each cluster. There is a degree of uncertainty of a pixel belonging to the different clusters. This uncertainty is higher at the cluster boundaries. Fuzzy values are assigned to every pixel to reflect its membership values to all the clusters.[18]

(a) *The ISODATA algorithm*

The following is a brief description of the fuzzy ISODATA algorithm of Bezdek and Dunn.[19] The description of the ordinary ISODATA algorithm is given in reference (19). Bezdek and Dunn[19] have shown that their fuzzy algorithm is superior to ordinary ISODATA from the standpoint of detecting the presence or absence of well-separated clusters in the data and of identifying such clusters if they exist.

A fuzzy c-partition of a set X is a c-tuple of functions $u_i(.) : X \rightarrow [0, 1]$, $0 \leq i \leq c-1$, which satisfies the condition $\Sigma_{i=0}^{c-1} u_i(x) - 1$ at each $x \in X$. u_i is called a fuzzy subset or fuzzy cluster in X. $u_i(x)$ is the grade membership of x in fuzzy set u_i.

The following steps summarize the fuzzy ISODATA algorithm:

Step 1: Choose a fuzzy partition $u_i(.)$ of c nonempty membership functions (c is a fixed integer between 2 and the number n, of elements in X). $u_i(.) \neq 0$, $0 \leq i \leq c-1$.

Step 2: Compute the c weighted means m_i,

$$m_i = \frac{\sum_{x \in X} (u_i(x))^2 . x}{\sum_{x \in X} (u_i(x))^2}, \quad 0 \leq i \leq c-1. \qquad (1)$$

These are the centers of clusters which are iteratively modified depending on the values of $u_i(.)$.

Step 3: Construct a new partition, $\hat{u}(.)$, as follows:

Let $I(x) = \{0 \leq i \leq c-1 \mid m_i = x\}$,
if $I(x)$ is not empty then

$$\hat{u}_i(x) = \begin{pmatrix} 1 & i = \hat{i} \\ 0 & i \neq \hat{i} \end{pmatrix}, \quad 0 \leq i \leq c-1 \qquad (2)$$

where $\hat{i}$ is the least integer in $I(x)$.
Otherwise, if $I(x)$ is empty (the usual case) then

$$\hat{u}_i(x) = \frac{(1/(\|x - m_i\|^2)}{\Sigma_{j=1}^{k} (1/\|x - m_j\|^2)}. \qquad (3)$$

Step 4: Compute some convenient measure, δ, of the defect between $u_i(.)$ and $\hat{u}_i(.)$. If δ is less than a specified threshold, ε, then stop; otherwise, put $u_i(.) = \hat{u}_i(.)$ and go to Step 2.

For simplicity, the ordinary Euclidean norm $\|.\|$ is used in the equations described above.

The ultimate goal of the clustering process is to segment each character into c disjoint clusters. Each $x \in X$ has different membership values in the c clusters. To minimize the risk, each pixel x is assigned to the cluster in which it has the maximum membership value.[19]

(b) *Character skeleton generation*

The clustering algorithm calculates the centers of clusters. These centers and the adjacency relations among the clusters are used to construct the skeleton of the characters.

The adjacency matrix, H, is a $c \times c$ matrix, where c is the number of clusters. The adjacency matrix, H, is determined by scanning the character image by a 2×2 window. If any two pixels of this window belong to two different clusters then these clusters are considered as adjacent.

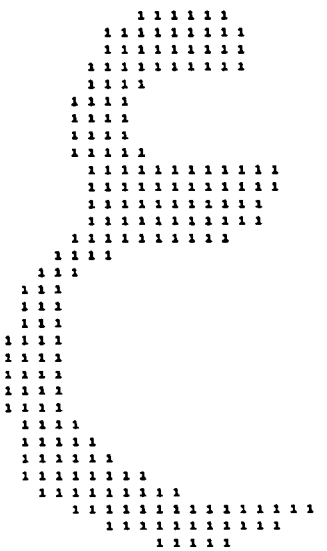


Fig. 2. The image of the Ain(ع) character.

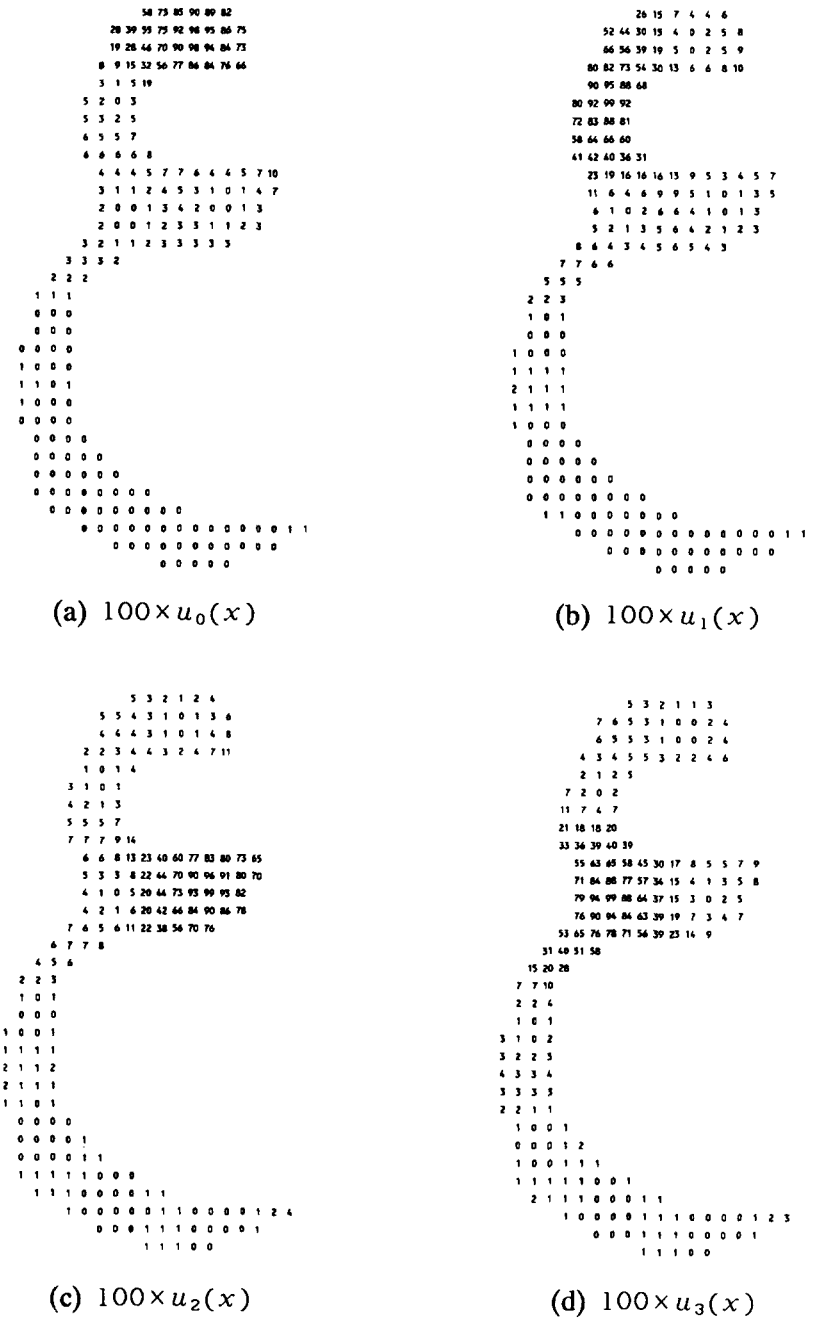


Fig. 3. (continued on facing page).

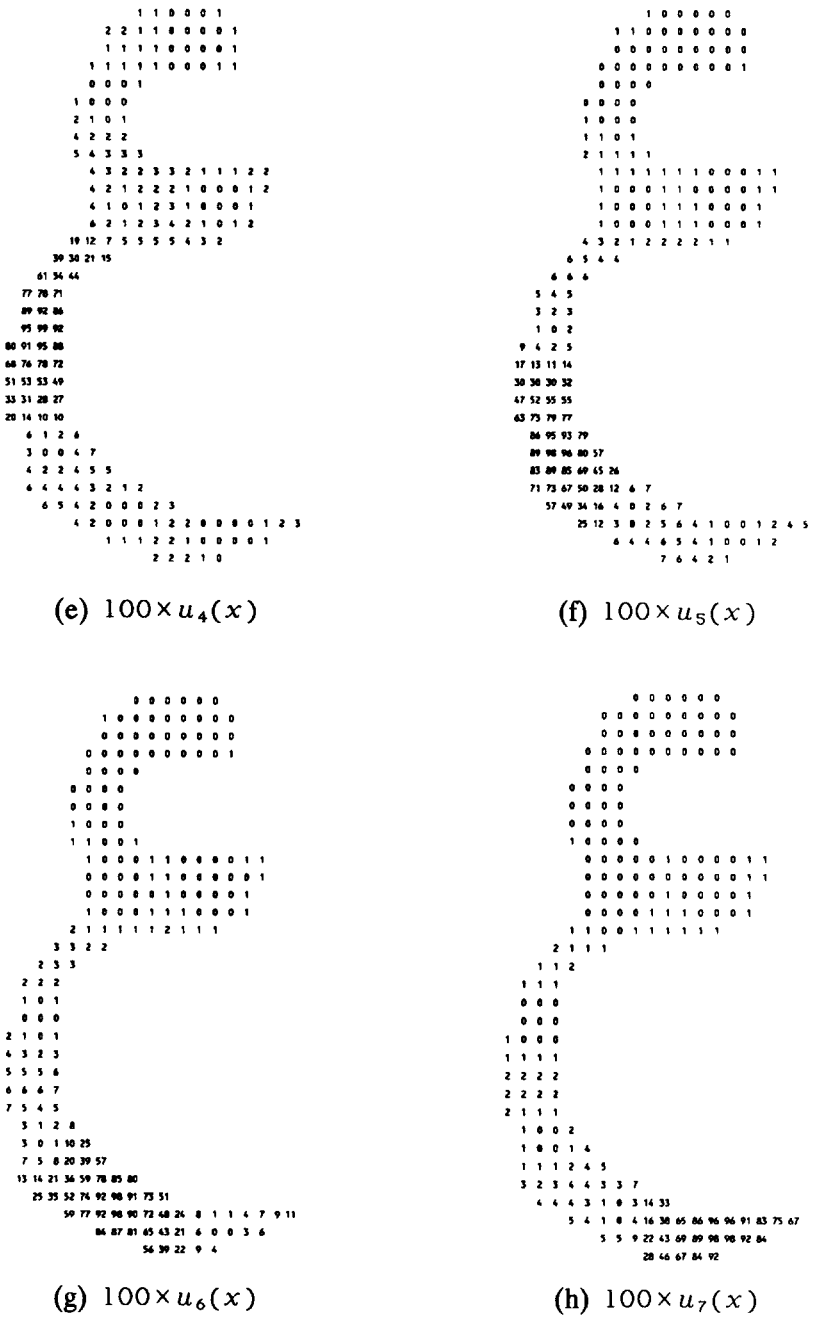
$h_{ij} = 1$ if clusters i and j are adjacent, $i \neq j$, and 0 otherwise.

The skeleton representing the character is obtained from the adjacency relation. Each cluster is represented by its center and is plotted as a single node. If any two clusters are adjacent then their corresponding nodes are connected by a straight line. The resulting graph is the skeleton of the character.

(c) *Initial membership assignment*

In the above fuzzy ISODATA algorithm, the c-tuple of functions, $u_i(.)$, are initialized. The effect of this initialization on the resulting clusters and on the processing time is analyzed using two initial-assignment strategies; the c-consecutive slices strategy (CCS) and the c-interlaced slices strategy (CIS).

(1) *The c-consecutive slices strategy.* In this strategy, the n elements of the X array are divided into c consecutive segments, S_i, $0 \leq i \leq c-1$. Each segment, S_i, initially contains an equal number of elements, except the last segment (as the number of black pixels, n, may not be a multiple integer of the



(e) $100 \times u_4(x)$ (f) $100 \times u_5(x)$

(g) $100 \times u_6(x)$ (h) $100 \times u_7(x)$

Fig. 3. The different fuzzy membership values of each pixel of the character image in the different clusters.

clusters number, c). The elements of each segment have full membership. So, if an element $x \in X$ is initially in S_i, then

$$u_j(x) = \begin{pmatrix} 1, & j = i \\ 0, & j \neq i \end{pmatrix}, \quad 0 \leq i, j \leq c - 1.$$

(2) *The c-interlaced slices strategy*. Here the cluster to which a black pixel x_k, $0 \leq k \leq n - 1$ is assigned is determined by finding the remainder of dividing the element number k by the number of clusters. The remainder is always less than c. So the cluster number equals k mod c. In this strategy the pixels are assigned to clusters in a semi-random fashion.

(d) *Skeleton reduction algorithm*

The following algorithm is applied to eliminate extra insignificant skeleton vertices (as a refining and

data-reduction step). Insignificant in the sense that there is near continuity in the direction between the segment entering and the one leaving the insignificant vertex.

Let a skeleton vertex, v_i, be connected to vertices v_j and v_k. The angle, α, between the two segments, S_{ij} and S_{ik}, is found. If α satisfies the condition $(180° - \gamma \leq \alpha \leq 180° + \gamma$, where γ is a threshold angle), then v_i is deleted from the adjacency matrix H (since the elimination of this vertex does not affect the general skeleton of the character). The entries of matrix H are modified such that $h_{il} = h_{li} = 0$, for all $l \in \{j, k\}$ and $h_{jk} = h_{kj} = 1$. This process is repeated until no more vertices can be eliminated. Experimental results have shown that $\gamma = 10°$ gives acceptable skeletons. The application of the skeleton reduction algorithm is expected to produce a considerable reduction in the number of vertices without introducing skeleton distortion. However, this result will be verified in the next section.

4. EXPERIMENTAL RESULTS

The following experiments were run on an IBM PS2 model 80, with 16 MHz clock and 80287 coprocessor. Row images are taken using an HP ScanJet scanner with Scangallary software. The resolution is 300 dpi (dots per inch), with 16 gray levels, and normal intensity. The above algorithms were implemented using C language. An IBM typing machine character set and handwritten characters are used in our experiments.

A binary image of a character is a matrix A, where a_{ij} is the matrix element in the ith row and the jth column, $1 \leq i \leq r$ and $1 \leq j \leq l$, where r is the number of rows, and l is the number of columns in A. a_{ij} is equal to 1 for black, and 0 for white. To reduce the processing time, the set of black points of the matrix A is transformed to an array X of n elements, where $n \leq r \times l$. Each element of X has two features (viz. the x- and the y-coordinates). After the row image is taken, an averaging and thresholding algorithm is applied on the image. The output is a binary image A, which is not necessarily a well-smoothed image. The set of black points of A are transformed into the 2-feature array X.

(a) *Clustering based skeletonization algorithm (CBSA)*

Figure 2 shows the Ain(ξ) character. Each black point is indicated by a 1, while white points are left blank. This letter is clustered, using the clustering algorithm of Section 2, into eight clusters. Figures 3(a)–(h) show the different membership values of each pixel of the character to each of the clusters i, $0 \leq i \leq 7$. It is noticed that, for each pixel $x \in X$, the equality $\Sigma_i^7 u_i(x) = 1$ holds true (i.e. summing the membership values for each pixel in Figs 3(a)–(h) is equal to unity). It is clear that the membership values increase as we go closer to the center of the cluster, and vice versa. Note that the membership values are multiplied by 100 to reduce the printout width. Figure 3(a) shows a cluster at the top of the character. The pixels which have large membership values in this cluster constitute the core of the cluster. Pixels which are far from the cluster core have small membership values. The same applies to other clusters. Figure 4 shows a fuzzy 8-partition of the character shown in Fig. 2 after applying the rule of maximum membership value. In this figure the clusters are labeled with decimal numbers from 0 to 7. The centers of clusters are indicated by capital "X". Fuzzy boundaries between fuzzy clusters are indicated by dotted lines. It is clear that, in general, adjacent pixels are assigned to the same cluster. This reflects the interesting behavior of the fuzzy ISODATA algorithm which tries to find the natural groupings of given data. These natural groupings are the clusters, or segments, of the character.

(b) *Initial assignment*

The Ain(ξ) character is initialized according to the CCS strategy, as shown in Fig. 5(a). Applying the fuzzy ISODATA algorithm and the rule of maximum membership, the final clusters shown in Fig. 5(b) are obtained. The same character is initially assigned using the CIS strategy as shown in Fig. 6(a). Figure 6(b) shows the final clusters. Although the initial assignments in Figs 5(a) and 6(a) are different, the final partitions are almost the same. This property of initial assignment independence is a very attractive one. For a given character, its images are not usually captured with the same orientation. Even if it is

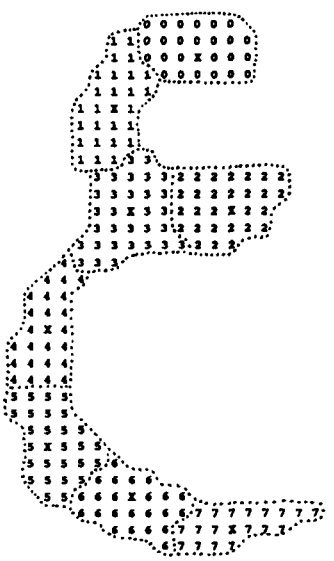


Fig. 4. The final clusters of the Ain(ξ) character.

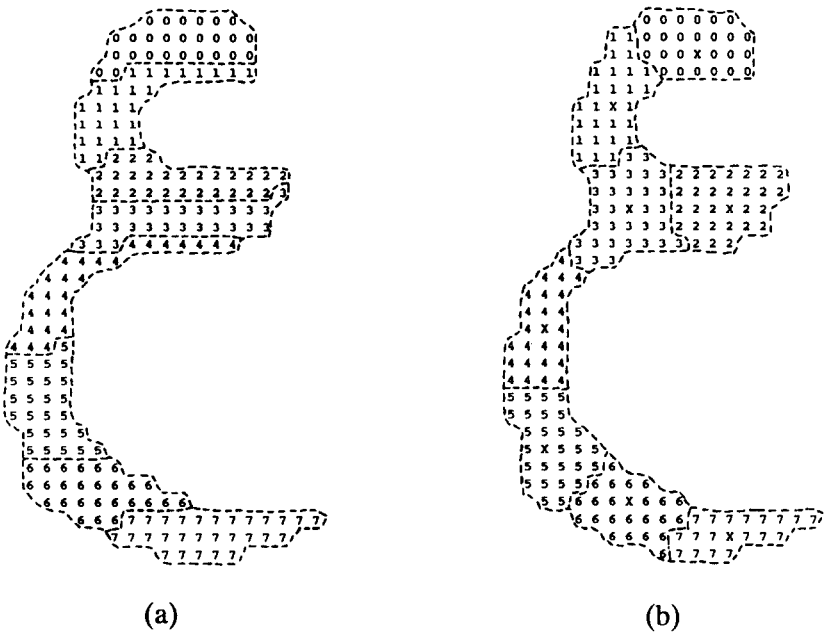


Fig. 5. (a) The initialization of clusters according to CCS strategy. (b) The final clusters of the Ain(ع) character.

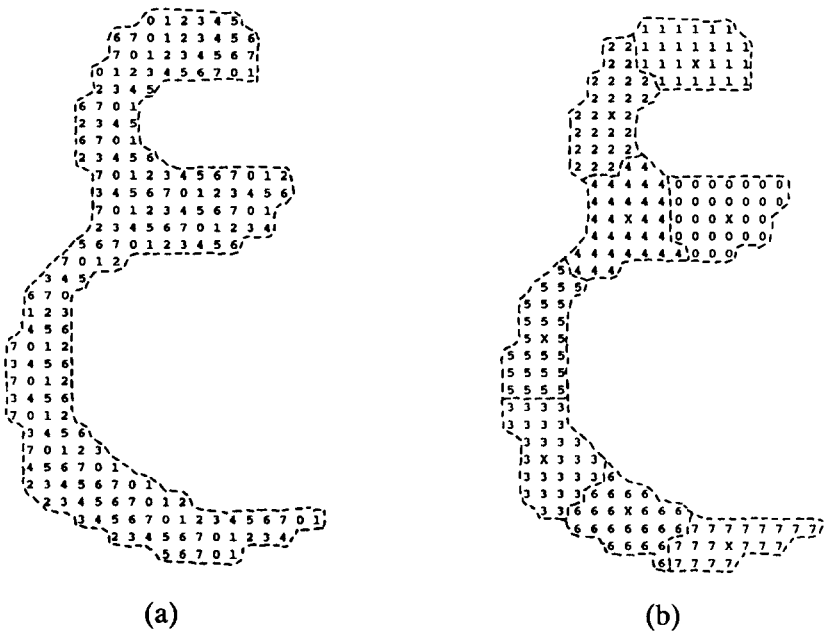
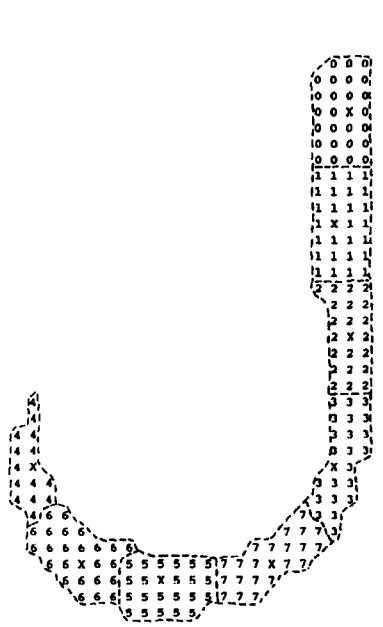


Fig. 6. (a) The initialization of clusters according to CIS strategy. (b) The final clusters of the Ain(ع) character.

captured with the same orientation, one cannot guarantee that identical images are obtained due to noise and rotation effects. Hence, it is difficult to obtain similar initial assignments for images referring to a given character. This explains the importance of the independence property of the initial assignment strategy in the clustering process. The processing time of the clustering algorithm is compared using the two initial assignment strategies under study. The running of the clustering algorithm using the CIS strategy is slower than that of the CCS strategy by nearly 50%. This is expected as in using the CCS strategy there is an initial grouping of clusters that saves in the processing time (that otherwise would be required from the clustering algorithm). Hence, the use of CCS strategy is recommended.

Figure 7(a) shows the final eight clusters for the Lam(ل) character. Figures 7(b) and (c) show the



(a)

		CLUSTER #							
		0	1	2	3	4	5	6	7
C	0	0	1	0	0	0	0	0	0
L	1	1	0	1	0	0	0	0	0
U	2	0	1	0	1	0	0	0	0
S	3	0	0	1	0	0	0	0	1
T	4	0	0	0	0	0	0	1	0
E	5	0	0	0	0	0	0	1	1
R	6	0	0	0	0	1	1	0	0
#	7	0	0	0	1	0	1	0	0

(b)

(c)

	CLUSTER#					
	0	1	2	3	4	5
C						
L 0	0	1	0	0	0	0
U 1	1	0	0	0	0	1
S 2	0	0	0	0	1	0
T 3	0	0	0	0	1	1
E 4	0	0	1	1	0	0
R 5	0	1	0	1	0	0
#						

(d)

(e)

Fig. 7. (a) The clusters of the Lam(ل) character. (b) The adjacency matrix of (a). (c) The skeleton of the character. (d) The refined adjacency matrix. (e) The refined skeleton of the character.

adjacency matrix and the skeleton of the character, respectively. Applying the skeleton reduction algorithm to the skeleton of the character gives the refined adjacency matrix of Fig. 7(d) and the refined skeleton of Fig. 7(e).

Figure 8(a) shows the images of four representative handwritten Arabic characters (viz. Ain(ع), Fa(ف), Ha(ه) and Ta(ط)). Figures 8(b)–(d) show the skeletons produced by CBSA, SPTA and Pavlidis skeletonization techniques, respectively. It is clear from the figures that the data reduction of CBSA is larger than the other techniques. The CBSA skeleton of Fig. 8(b) represents the general structure relationships of the characters in a more natural way, with no extra spur tails nor breaking of continuous line segments into smaller segments (that are stair-case or zigzag-like) as is the case with Figs 8(c) and (d). The dot of the Fa(ف) character is represented by a 1-pixel dot using CBSA which is the representation we would like to get for the dot of the character. The SPTA thinned the dot into two 1-pixel dots and Pavlidis algorithm eliminated the dot. The Fa(ف) skeleton with two 1-pixel dots will be recognized as Qaf(ق) character and the elimination of the dot produces an invalid character. Hence, these two techniques suffer from this limitation. In addition, the skeletons of SPTA and Pavlidis have a concavity at the junction of the Fa(ف) loop with the main skeleton. This concavity is not present in the character. The concavity is more pronounced in the SPTA skeleton which confirms the expected behavior of this algorithm at "T" junctions which are skeletonized to more of a "Y" than a "T". However, the general skeletons

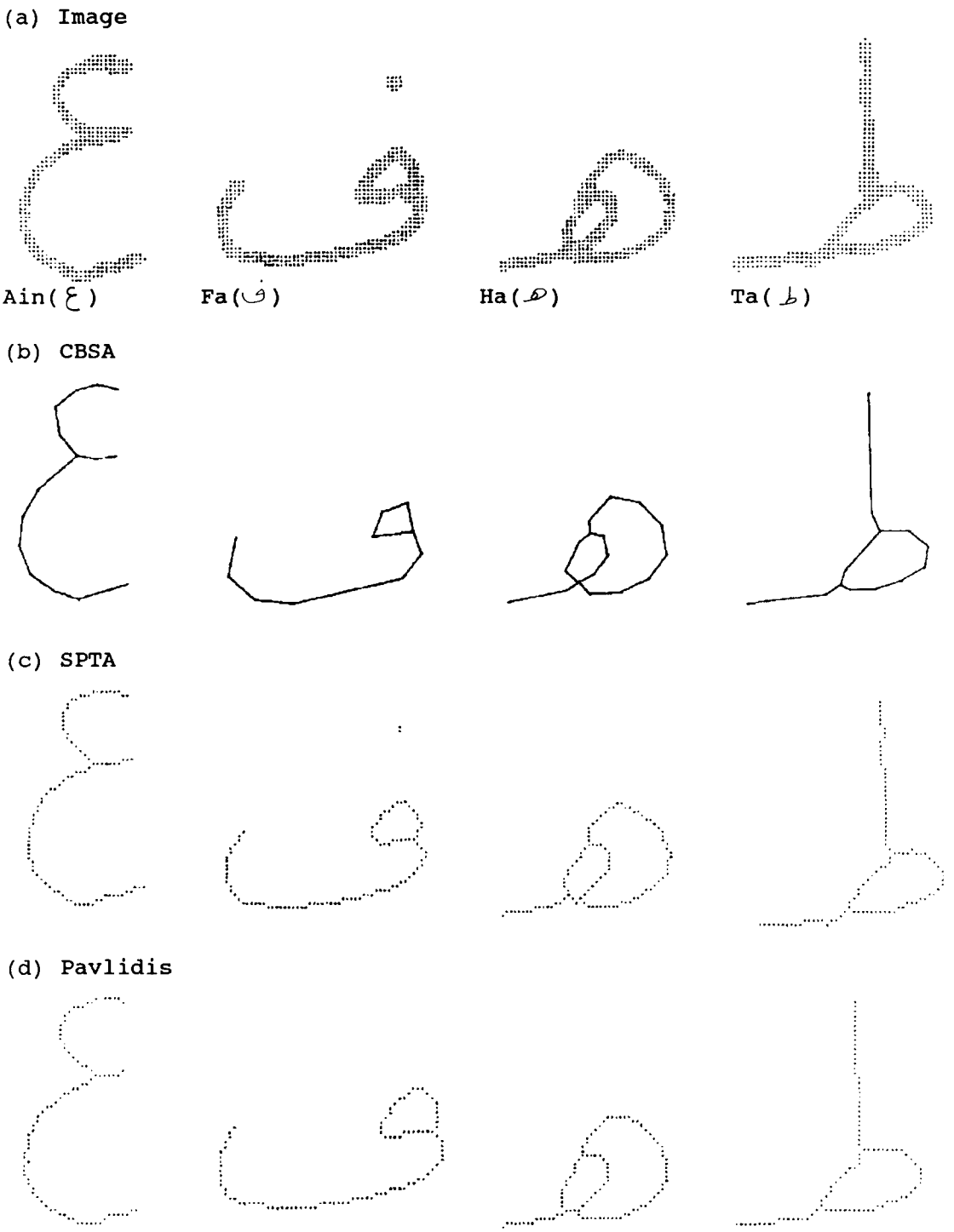


Fig. 8. (a) The original images of Ain(ع), Fa(ف), Ha(ه), and Ta(ط) characters; (b, c, d) CBSA, SPTA, Pavlidis skeletons of the characters of (a), respectively.

are similar otherwise. The above observations indicate the superiority of the CBSA algorithm. However, the algorithm requires more processing time than other techniques.

(c) *Noise effects*

Images of a character may suffer from the effects of noise. A robust skeleton-generating algorithm produces skeletons which reflect the general character structure irrespective of noise.

Noise is a natural phenomenon, which may be generated by the equipment used to capture the image, due to quantization errors, etc. Pictures are first recorded as gray level images which are then

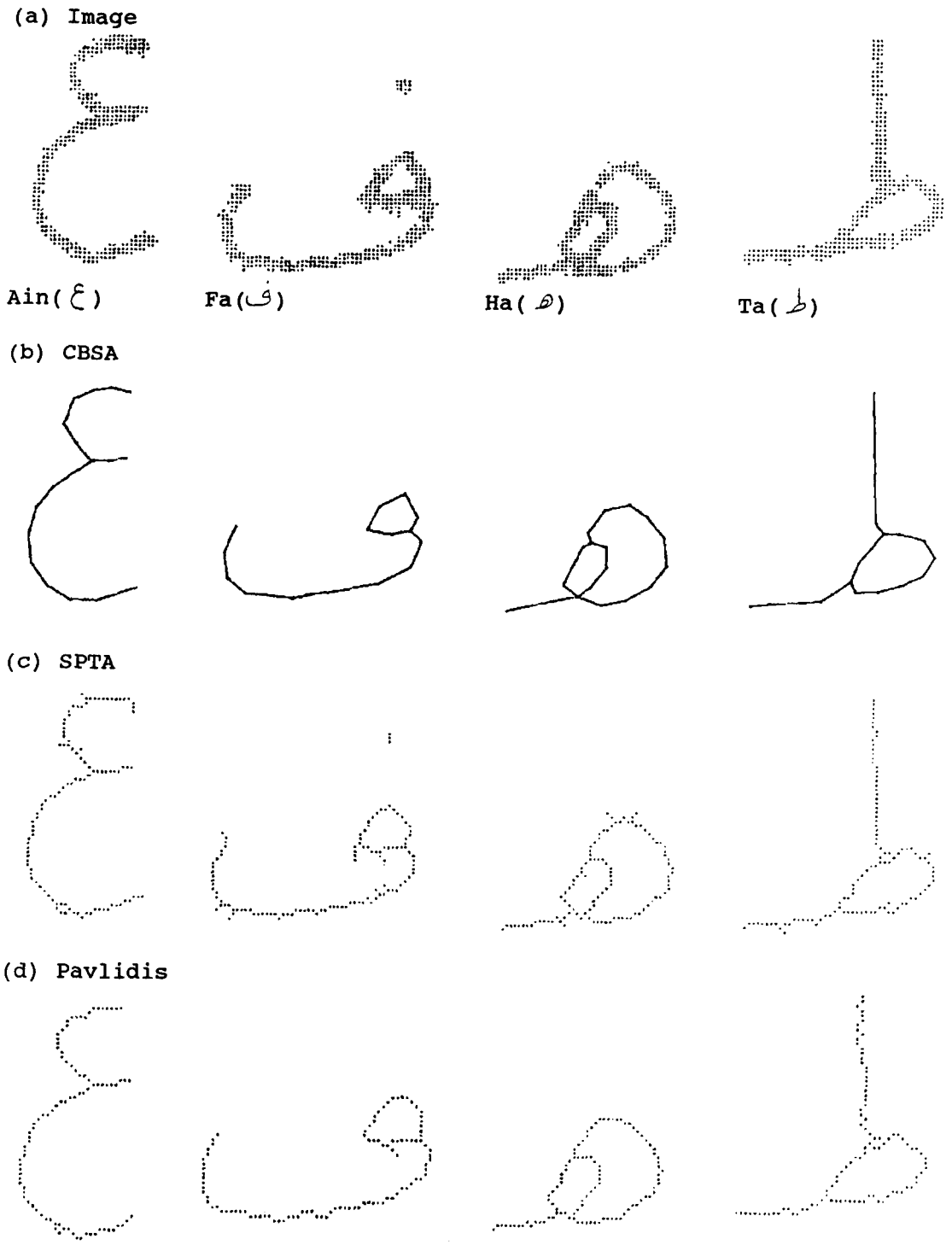


Fig. 9. (a) The original images of Ain(ع), Fa(ف), Ha(ه), and Ta(ط) characters with additive noise; (b, c, d) CBSA, SPTA, Pavlidis skeletons of the characters of (a), respectively.

converted to binary images for recognition purposes. This conversion from gray to binary introduces noise to the image. The overall pepper and salt noise is easily removed by averaging or by eliminating isolated black pixels, hence it will not be addressed here. Random additive noise is applied at and near the character edges. This noise is common in typewritten text due to print-wheel wear and ribbon ink. The effect of this type of noise is analyzed below.

Figure 9(a) shows the characters of Fig. 8(a) but with the addition of random noise at the edges. Figures 8(b)–(d) show the skeletons of the characters

using CBSA, SPTA, and Pavlidis algorithms, respectively. The skeletons of Figs 8(c) and (d) have more spur tails, more stair-case or zigzag-like lines, and small loops, as is clear from the figures. The dot in the Fa(ف) character is represented by three 1-pixel dots in the SPTA skeleton and is eliminated in the other. The "T" junction of the two skeletons is worse in this case. The "T" connections of the Ha(ـهـ) and Ta(ـط) of the CBSA skeletons are slightly more concave than before the addition of noise. Again, the CBSA skeletons are superior to others. Although Pavlidis algorithm eliminated the dot in this case, there are other cases where it preserves the dot in a similar fashion to SPTA.

5. CONCLUSIONS

A clustering based skeletonization algorithm (CBSA) for Arabic characters is presented. CBSA has the advantages of other thinning algorithms in reducing the data, simpler data structure, and in making further processing of patterns less time-consuming as only a small percentage of data is processed. In addition, it enjoys other advantages compared to other thinning algorithms. The presented technique is robust as it is less sensitive to noise. While other thinning algorithms produce skeletons that are not of equal number of vertices for different characters, the proposed algorithm can produce the same number of vertices by keeping the number of clusters fixed and not applying the refining algorithm. This property is useful as it is suitable for character recognition techniques using parallel algorithms. For example, in using neural networks for the recognition algorithms, the data input length should be fixed. This is possible using the presented skeletonization algorithm while this is not the case using other thinning algorithms. However, the presented algorithm is time-consuming and hence is suitable for patch processing. The total recognition system may be slow for real time applications. To improve the performance of the overall system, the above skeletonization algorithm is applied in the modeling and training stages and other thinning algorithms may be used in the recognition stage.

The experimental results indicate that different characters are better clustered using different numbers of clusters to generate an adequate skeleton and save in processing time. Research is in progress to formulate an algorithm that estimates the suitable number of clusters for a given character. The recognition of Arabic characters using CBSA algorithm and neural networks, which has the advantages of parallelism and learning, is under study. Comparable or even better results than other recognition techniques of Arabic text are expected as was the case in applying neural networks to English character recognition.[21]

SUMMARY

This paper presents a skeletonization algorithm for Arabic characters. The included characteristics of Arabic text indicate that skeletonization techniques of other languages (for example, Latin or Chinese) are not readily applicable to Arabic characters. The skeletonization of Arabic characters required special algorithms to address the complications of Arabic text.

The clustering based skeletonization algorithm (CBSA) is based on clustering the character image. An adjacency matrix of the character cluster centers is found by scanning the clustered image by a 2×2 window. Two clusters are considered adjacent if each has at least one pixel in the window. The character skeleton is generated from the adjacency matrix. A skeleton-refining algorithm is then applied, which eliminates insignificant vertices (i.e. vertices whose removal does not alter the main structure of the character). The performance of two initial assignment strategies (viz. the c-consecutive slices strategy, CCS, and the c-interlaced slices strategy, CIS) is analyzed. The CCS initial assignment strategy is found to be superior to the CIS strategy in terms of processing time while both have the same cluster partitions of the character. Finally, CBSA, SPTA and Pavlidis skeletons are compared.

The experimental results show that CBSA has the advantages of other thinning algorithms in reducing the data, simpler data structure, and in making further processing of patterns less time-consuming as only small percentage of data is processed. In addition, CBSA is robust as it is less sensitive to noise. While other thinning algorithms produce skeletons that are not of equal number of vertices for different characters, the proposed algorithm can produce the same number of vertices by keeping the number of clusters fixed and not applying the refining algorithm. However, the presented algorithm is time-consuming and hence is suitable for patch processing. The total recognition system may be slow for real time applications. To improve the performance of the overall system, the above skeletonization algorithm is applied in the modeling and training stages and other thinning algorithms may be used in the recognition stage.

REFERENCES

1. Pepi Siy and C. S. Chen, Fuzzy logic for handwritten numeral character recognition, *IEEE Trans. Syst. Man Cybern.* 570–575 (1974).
2. W. J. M. Kickert and H. Koppelaar, Application of fuzzy set theory to syntactic pattern recognition of handwritten capitals, *IEEE Trans. Syst. Man Cybern.* 148–151 (1976).
3. Ching Y. Suen, M. Berthod and Shunji Mori, Automatic recognition of handprinted characters—the state of the art, *Proc. IEEE* **68**, 469–487 (1980).

4. B. Parhami and M. Tapaghi, Automatic recognition of printed Farsi texts, *Pattern Recognition* **14**, 395–403 (1981).
5. G. Wolberg, A syntactic omni-font character recognition, *Proc. IEEE Conf. Comput. Vision Pattern Recognition*, Miami Beach, FL, pp. 168–173, 22–26 June (1986).
6. M. Beun, A flexible method for automatic reading of handwritten numerals, *Phillips tech. Rev.* **33**, (5), 130–137 (1973).
7. H. Almuallim and S. Yamaguchi, A method of recognition of Arabic cursive handwriting, *IEEE Trans. Pattern Anal. Mach. Intell.* **PAMI-9**, (5) (1987).
8. S. Kahan, T. Pavlidis and H. S. Baird, On the recognition of printed characters of any font and size, *IEEE Trans. Pattern Anal. Mach. Intell.* **PAMI-9**, (2) (1987).
9. T. Pavlidis, *Algorithms for Graphics and Image Processing*, pp. 195–214. Computer Science Press, Rockville, MD (1982).
10. G. P. Dinneen, Programming pattern recognition, *West jt Comput. Conf.* 94–100 (1955).
11. A. Rosenfeld and J. Pfaltz, Sequential operations in digital picture processing, *J. Ass. comput. Mach.* **13**, 471–494 (1966).
12. C. J. Hilditch, Linear skeletons from square cupboards, *Mach. Intell.* **4**, 403–420 (1969).
13. K. J. Udupa and I. S. N. Murthy, Some new concepts for encoding line patterns, *Pattern Recognition* **7**, 225–233 (1975).
14. N. J. Naccache and R. Shinghal, SPTA: A proposed algorithm for thinning binary patterns, *IEEE Trans. Syst. Man Cybern.* **SMC-14** (3), 409–418 (1984).
15. Sabah AbdulAziz Ali and Mahdi Saleh AlSaadon, Parallel algorithm for thinning images (in Arabic), *First Kuwait Comput. Conf.*, Kuwait, pp. 121–140 (1989).
16. Hani M. K. Mahdi, Thinning and transforming the segmented Arabic characters into meshes of unified small size, *Proc. 14th Int. Conf. Statist., Comput. Sci. Demographic Res.*, Cairo, Egypt, pp. 263–276, 25–30 March (1989).
17. A. Nouh, N. Ula and A. Sharaf Eldin, A proposed algorithm for thinning binary Arabic character patterns, *First Kuwait Comput. Conf.*, Kuwait, pp. 426–441 (1989).
18. L. Zadeh, Fuzzy sets, *Inf. Control* **8**, 338–353 (1965).
19. J. Bezdek and J. Dunn, Optimal Fuzzy partitions: A heuristic for estimating the parameters in a mixture of normal distributions, *IEEE Trans. Comput.* **C-24**, 835–838 (1975).
20. J. Bezdek and P. Castelaz, Prototype classification and feature selection with fuzzy sets, *IEEE Trans. Syst. Man Cybern.* **SMC-7** (2), 87–92 (1977).
21. A. Rajavelu, M. T. Musavi and M. V. Shirvaikar, A neural network approach to character recognition, *Neural Networks* **2**, 387–393 (1989).

About the Author—SABRI A. MAHMOUD received his B.S. in Electrical Engineering from Sind University, Pakistan, in 1972. He was an Electrical Engineer for five years at Shuaiba North Power Station, Kuwait. He received his M.S. in Computer Sciences from Stevens Institute of Technology, U.S.A., in 1980. He has worked with Royal Saudi Naval Forces as senior systems engineer for five years. He received his Ph.D. in Information Systems Engineering from the University of Bradford, U.K., in 1987.

He is currently an Assistant Professor at the Computer Engineering Department, College of Computers and Information Systems, King Saud University, Saudi Arabia. His research interests include image processing, pattern recognition and the use of fuzzy concepts and neural networks in the above applications.

Dr Mahmoud is a senior member of IEEE and a member of Acoustic, Speech, and Signal Processing Society.

About the Author—IBRAHIM S. ABUHAIBA received his B.Sc. (Honors) degree from Systems and Computer Engineering Department, AlAzhar University, Cairo, Egypt, in May 1984. Since October 1984 he has been a research assistant at the Computer Engineering Department, College of Computers and Information Systems, King Saud University, Riyadh, Saudi Arabia. His research interests include character recognition, computer communications, robotics and neural networks.

Mr AbuHaiba is a member of IEEE.

About the Author—ROGER J. GREEN obtained his B.Sc. in Electronics from Manchester University in 1973 and his Ph.D. in Communications from Bradford University in 1976. After several years in industry, he worked at Leeds University as a lecturer, and moved to Bradford University in 1981, and is currently a Senior lecturer. His research interests include imaging systems, electro-optics and optoelectronics. He is an IEE member, and a Senior Member of the IEEE. He is the holder of a patent on low noise optical detection, and has published widely.